

3. Introduction to mCRL2

José Proença

System Verification (CC4084) 2024/2025

CISTER – U.Porto, Porto, Portugal

<https://fm-dcc.github.io/sv2425>



CISTER - Research Centre in
Real-Time & Embedded
Computing Systems

`http://mcr12.org`

- Formal [specification language](#) with an associated toolset
- Used for [modelling](#), [validating](#) and [verifying](#) concurrent systems and protocols
- Tool suggestion: use [mcr12ide](#) (not `mcr12-gui`)

$$\begin{array}{c}
 \text{(act)} \\
 \hline
 \alpha.P \xrightarrow{\alpha} P
 \end{array}
 \qquad
 \begin{array}{c}
 \text{(sum-1)} \\
 \hline
 \frac{P_1 \xrightarrow{\alpha} P'_1}{P_1 + P_2 \xrightarrow{\alpha} P'_1}
 \end{array}
 \qquad
 \begin{array}{c}
 \text{(sum-2)} \\
 \hline
 \frac{P_2 \xrightarrow{\alpha} P'_2}{P_1 + P_2 \xrightarrow{\alpha} P'_2}
 \end{array}$$

$$\begin{array}{c}
 \text{(res)} \\
 \hline
 \frac{P \xrightarrow{\alpha} P'}{P \setminus L \xrightarrow{\alpha} P' \setminus L} \quad \alpha \notin L
 \end{array}
 \qquad
 \begin{array}{c}
 \text{(rel)} \\
 \hline
 \frac{P \xrightarrow{\alpha} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]}
 \end{array}$$

$$\begin{array}{c}
 \text{(com1)} \\
 \hline
 \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q}
 \end{array}
 \qquad
 \begin{array}{c}
 \text{(com2)} \\
 \hline
 \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}
 \end{array}
 \qquad
 \begin{array}{c}
 \text{(com3)} \\
 \hline
 \frac{P \xrightarrow{a} P' \quad Q \xrightarrow{\bar{a}} Q'}{P|Q \xrightarrow{\tau_a} P'|Q'}
 \end{array}$$

Processes in mCRL2

Syntax (by example)

$$a.\mathbf{0} \rightarrow a$$

$$a.P \rightarrow a.P$$

$$P_1 + P_2 \rightarrow P_1 + P_2$$

$$P \setminus L \rightarrow \text{block}(L, P)$$

$$P[f] \rightarrow \text{rename}(f, P)$$

$$a.P \mid \bar{a}.Q \rightarrow \text{comm}(\{a_1 \mid a_2 \rightarrow a\}, a_1.P \mid a_2.Q)$$

$$a.P \mid \bar{a}.Q \setminus \{a\} \rightarrow \text{block}(\{a_1, a_2\}, \text{comm}(\{a_1 \mid a_2 \rightarrow a\}, a_1.P \mid a_2.Q))$$

Syntax (by example)

$$a.0 \rightarrow a$$

$$a.P \rightarrow a.P$$

$$P_1 + P_2 \rightarrow P_1 + P_2$$

$$P \setminus L \rightarrow \text{block}(L, P)$$

$$P[f] \rightarrow \text{rename}(f, P)$$

$$a.P \mid \bar{a}.Q \rightarrow \text{hide}(\{a\}, \text{comm}(\{a1 \mid a2 \rightarrow a\}, a1.P \mid a2.Q))$$

$$a.P \mid \bar{a}.Q \setminus \{a\} \rightarrow \text{hide}(\{a\}, \text{block}(\{a1, a2\}, \text{comm}(\{a1 \mid a2 \rightarrow a\}, a1.P \mid a2.Q)))$$

$$CM = \text{coin}.\overline{\text{coffee}}.CM$$

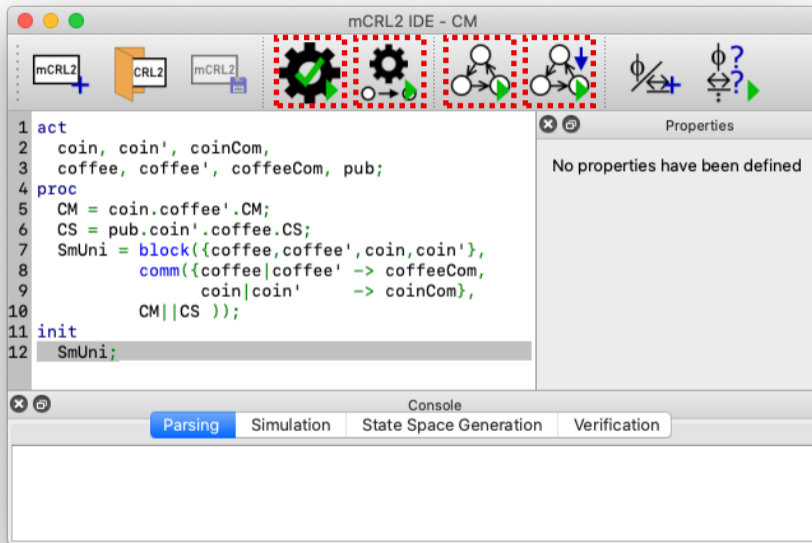
$$CS = \text{pub}.\overline{\text{coin}}.\text{coffee}.CS$$

$$SmUni = (CM|CS)\setminus\{\text{coin}, \text{coffee}\}$$

```

act
  coin, coin', coinCom,
  coffee, coffee', coffeeCom, pub;
proc
  CM = coin.coffee'.CM;
  CS = pub.coin'.coffee.CS;
  SmUni = block({coffee, coffee', coin, coin'},
    comm({coffee|coffee' -> coffeeCom,
          coin|coin'      -> coinCom},
    CM||CS ));
init
  SmUni;

```



The screenshot shows the mCRL2 IDE interface. The main window is titled "mCRL2 IDE - CM". The toolbar contains several icons, with a red dashed box highlighting a group of icons: a gear with a green checkmark, a gear with a green drop, a state transition diagram with a green checkmark, and a state transition diagram with a blue arrow pointing to a state. Below the toolbar is a code editor with the following code:

```

1 act
2   coin, coin', coinCom,
3   coffee, coffee', coffeeCom, pub;
4 proc
5   CM = coin.coffee'.CM;
6   CS = pub.coin'.coffee.CS;
7   SmUni = block({coffee,coffee', coin, coin'},
8               comm({coffee|coffee' -> coffeeCom,
9                   coin|coin'       -> coinCom}),
10              CM||CS ));
11 init
12   SmUni;

```

To the right of the code editor is a "Properties" panel with the text "No properties have been defined". Below the code editor is a "Console" panel with tabs for "Parsing", "Simulation", "State Space Generation", and "Verification". The "Parsing" tab is currently selected.

Parse

Simulate

Visualize

Minimize &

Visualize


```
act
  action1, action2, ...;
  action3, action4 : Type;

proc
  P1 = ...;
  P2(x: Bool) = ...;
    % Process expression

init
  SmUni;
```

```
sort List = struct
  empty | cons(A,List);

map sum2: Int # Int -> Int;

var x, y: Int;

eqn
  sum2(x,y) = (x+y) * (x+y);
  % Data patterns & expressions
```

https://mcr12.org/web/user_manual/language_reference/index.html

$$P = PE ;$$

a *Action*

a|b *Multi-action*

P *Process*

delta *Deadlock*

a(DataExpr) *Parameterized Act.*

P(DataExpr) *Parameterized Proc.*

a.PE *Sequencing*

PE1 + PE2 *Choice*

PE1 || PE2 *Parallel*

block({a,b},PE) *Block*

allow({a,b},PE) *Allow*

rename({a->b},PE) *Rename*

comm({a|b->c},PE) *Communicate*

sum m: Nat . PE *Gen. Choice*

$P(\text{exp})$

true *Boolean*

42 *Pos, Nat, Int, Real*

exp! *Not*

exp && exp *And*

exp || exp *Or*

exp => exp *Implies*

forall n:Nat . exp *For all*

exists n:Nat . exp *Exists*

exp + exp *Sum*

max(exp, exp) *And*

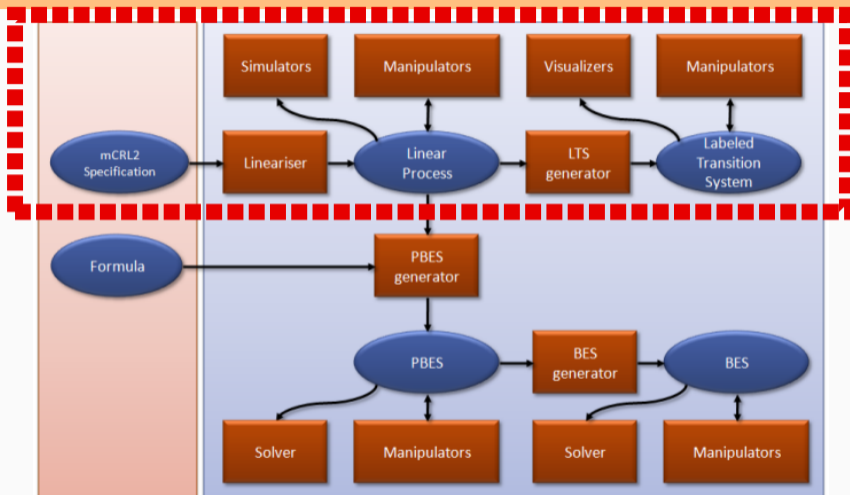
exp mod exp *Remainder of div.*

[exp, exp, ...] *List*

{exp, exp, ...} *Set*

{exp:2, exp:1, ...} *Bag*

lambda n:Nat . exp *Function*



Assignment 1: <https://fm-dcc.github.io/sv2425/exercises/mcrl2-assignment.zip>

```
https://dcc-fm.github.io/sv2425/exercises/  
adventurers/adventurers-tutorial-mcrl2.zip
```

Logic and Verification

The screenshot shows the mCRL2 IDE interface. The main window is titled "mCRL2 IDE - CM". The top toolbar contains several icons: a plus sign, a folder icon labeled "CRL2", a gear with a checkmark, a gear, a state transition diagram, a state transition diagram with a blue arrow, and two red dashed boxes containing mathematical symbols (a crossed-out ϕ and a ϕ with a question mark). The code editor displays the following code:

```
1 act
2   coin, coin', coinCom,
3   coffee, coffee', coffeeCom, pub;
4 proc
5   CM = coin.coffee'.CM;
6   CS = pub.coin'.coffee.CS;
7   SmUni = block({coffee,coffee', coin, coin'},
8               comm({coffee|coffee' -> coffeeCom,
9                   coin|coin' -> coinCom},
10                  CM||CS ));
11 init
12   SmUni;
```

The Properties panel on the right is titled "Properties" and contains the text "No properties have been defined". The Console panel at the bottom is titled "Console" and has tabs for "Parsing", "Simulation", "State Space Generation", and "Verification". The "Parsing" tab is currently selected.

Add
properties

Verify
properties

Syntax (simplified)

$$\phi = \text{true} \mid \text{false} \mid \text{forall } x:T.\phi \mid \text{exists } x.:T\phi \\ \mid \phi \text{ OP } \phi \mid !\phi \mid [\text{expr}]\phi \mid \langle \text{expr} \rangle \phi \mid \dots$$

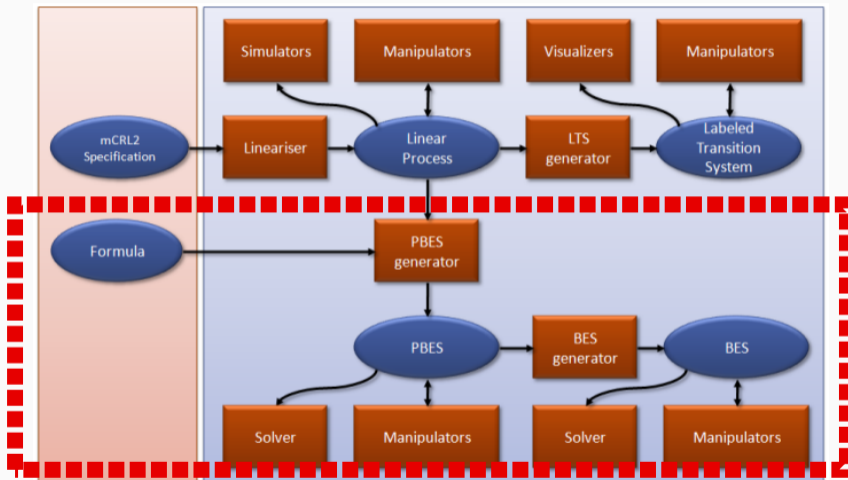
$$\text{expr} = \alpha \mid \text{nil} \mid \text{expr}+\text{expr} \mid \text{expr}.\text{expr} \mid \text{expr}^* \mid \text{expr}+$$

$$\alpha = a(d) \mid a|b|c \mid \text{true} \mid \text{false} \mid \alpha \text{ OP } \alpha \mid !\alpha \\ \mid \text{forall } x:T.\alpha \mid \text{exists } x:T.\alpha \mid \dots$$

where $T = \{Bool, Nat, Int, \dots\}$ and $OP = \{=\!, \&\&, \mid\mid\}$

Example

“ $[\text{true}^*.a]\langle b \rangle \text{true}$ ” means: *whenever an ‘a’ appears after any number of steps, it must be immediately followed by ‘b’.*



Assignment 1 (now everything):

<https://fm-dcc.github.io/sv2425/exercises/mcrl2-assignment.zip>